Using Really Big Computers to Study Rather Small Nuclei

Steven C. Pieper

Physics Division, Argonne National Laboratory

Work with

Ralph Butler, Middle Tennessee State U.

Anthony Chan, Argonne

Rusty Lusk, Argonne

Joseph Carlson, Los Alamos

Bob Wiringa, Argonne

Work not possible without extensive computer resources:

DOE INCITE access to Argonne's Blue Gene/P

Argonne Laboratory Computing Resource Center (Jazz, Fusion)

Argonne Math. & Comp. Science Division (SiCortex)

NERSC IBM SP's (Seaborg, Bassi)

U.S. DEPARTMENT OF ENERGY

Office of Science

Office of Nuclear Physics

Exploring Nuclear Matter - Quarks to Stars

Argonne NATIONAL LABORATORY Physics Division

U.S. DEPARTMENT OF ENERGY INCITE LEADERSHIP COMPUTING

SciDAC Scientific Discovery through Advanced Computing

Microscopic Few- & Many-Nucleon Calculations

Goal: a microscopic description of nuclear structure and reactions from bare NN & $3N$ forces.

There are two problems that must be solved to obtain this goal

(I) What is the Hamiltonian (i.e. the nuclear forces)?

- NN force controlled by NN scattering – lots of data available
  - Argonne v_{ij}
- $3N$ force determined from properties of light nuclei
  - Recent Illinois models with 2π & 3π rings

(II) Given H, solve the Schrödinger equation for A nucleons accurately.

- Essential for comparisons of models to data
- Quantum Monte Carlo has made much progress for $A \leq 12$
- Nuclei go up to A=238 and beyond!
  – less accurate approximations are used beyond 12

Without (II) comparison to experiment says nothing about (I).

ACCURATE REPRESENTATIONS OF NUCLEAR FORCES

- 1935: Meson-exchange theory of Yukawa
- 1953: Δ (33) resonance discovered by Anderson & Fermi
- 1955: Fujita-Miyazawa three-nucleon potential based on Δ excitation
- 1957: First phase-shift analysis of NN scattering data
- 1957–1968: Gammel-Thaler, Hamda-Johnston & Reid phenomenological potentials
- 1970s: Bonn, Nijmegen & Paris field-theoretic models
- 1993: Nijmegen Partial Wave Analysis (PWA93) $\rightarrow \chi^2 \sim 1$
- 1993–1996: Nijm I, Nijm II, Reid93, Argonne v_{18} & CD-Bonn
- 2004: Effective field theory at N^{3}LO

NUCLEAR HAMILTONIAN

$$H = \sum_i K_i + \sum_{i<j} v_{ij} + \sum_{i<j<k} V_{ijk}$$

K_i: Non-relativistic kinetic energy, $m_n - m_p$ effects included

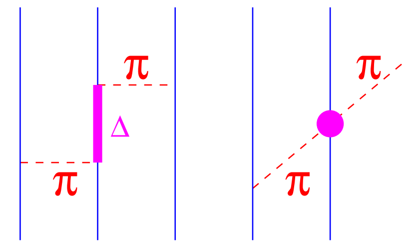
v_{ij}: Argonne v18 (1995)

- AV18 is a direct fit to 4300 NN data in the Nijmegen data base: χ^2/d.o.f. = 1.09

$$V_{ijk} = V_{ijk}^{2\pi} + V_{ijk}^{3\pi} + V_{ijk}^{R}$$

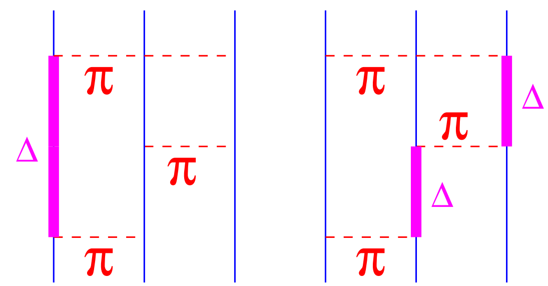
$V_{ijk}^{2\pi}$: Fujita-Miyazawa + s-wave term; in most V_{ijk}

- Longest ranged V_{ijk}
- Attractive in all nuclei studied.



$V_{ijk}^{3\pi}$: 3π rings with Δ's; new in Illinois V_{ijk}

- Extra p-shell, $|N - Z|$ attraction
- $\langle V_{ijk}^{3\pi} \rangle \lesssim 0.1 \langle V_{ijk}^{2\pi} \rangle$



In light nuclei we find $\langle V_{ijk} \rangle \sim (0.02 \text{ to } 0.09) \langle v_{ij} \rangle \sim (0.15 \text{ to } 0.6) \langle H \rangle$

(Large cancellation of K and v_{ij})

The Many-body Problem

Need to solve the Schrödigner Equation for A nucleons:

$$H\Psi(\vec{r}_1, \vec{r}_2, \cdots, \vec{r}_A; s_1, s_2, \cdots, s_A; t_1, t_2, \cdots, t_A)$$
$$= E\Psi(\vec{r}_1, \vec{r}_2, \cdots, \vec{r}_A; s_1, s_2, \cdots, s_A; t_1, t_2, \cdots, t_A)$$

s_i are nucleon spins: $\pm\frac{1}{2}$

t_i are nucleon isospins (proton or neutron): $\pm\frac{1}{2}$

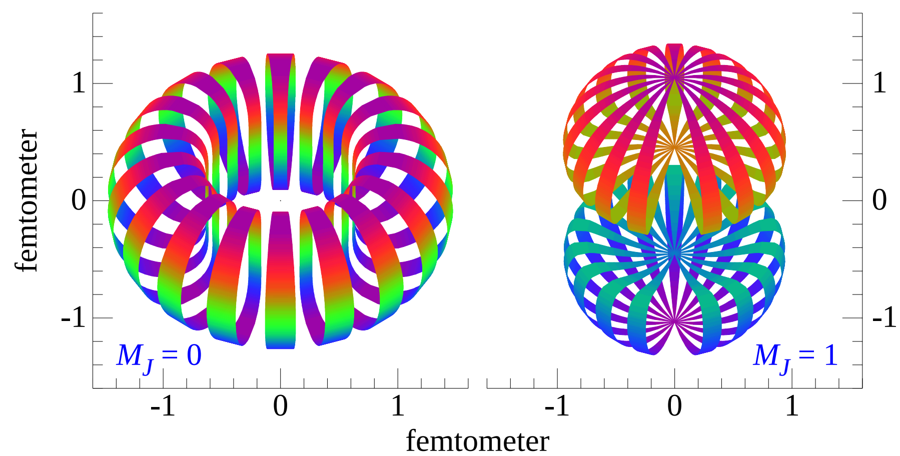
$2^A \times \binom{A}{Z}$ complex coupled 2^{nd} order eqn in $3A - 3$ variables

(number of isospin states can be reduced)

^{12}C: 270,336 coupled equations in 33 variables

Coupling is strong:

- $\langle v_{\text{Tensor}} \rangle$ is $\sim 60\%$ of total $\langle v_{ij} \rangle$
- $\langle v_{\text{Tensor}} \rangle = 0$ if no tensor correlations



Accurate Solutions of Many-Body Schrödinger Equation

- ^{2}H by Numerical Integration (1952)
  –"5 to 20 minutes for calculation and 10 minutes to print result"

- 1981: Lomnitz-Adler, Pandharipande & Smith – 1^{st} nuclear Variational Monte Carlo
  – ^{3}H & ^{4}He using the Reid NN potential

- 1987: Carlson – 1^{st} nuclear Green's function Monte Carlo – ^{3}H & ^{4}He with v_6 potential
- 1987: Carlson, Schmidt & Kalos – VMC calculation of n–^{4}He scattering phase shifts
- 1988: Carlson – GFMC for ^{3}H & ^{4}He with full Reid v_8 potential (tensor and $L \cdot S$ terms)
- 1991: Carlson – GFMC calculation of n–^{4}He scattering phase shifts (large statistical errors)
- 1992: Pieper, Wiringa & Pandharipande – Cluster VMC calculation of ^{16}O
- 1995: Pudliner, Pandharipande, Carlson & Wiringa – GFMC for ^{6}He, ^{6}Li with AV18+UIX
- 1996–present: Slow but steady progress of GFMC to bigger nuclei (now at ^{12}C)
- 1995–present: No Core Shell Model up to ^{16}O
- 2001: ^{4}He benchmark by 7 methods to 0.1% (17 theorists on one paper!)
- 2005: ^{16}O by Coupled Cluster

Quantum Monte Carlo

Need to solve the Schrödigner Equation for A nucleons: $H\Psi_0 = E\Psi_0$
Ψ_0 is the "ground-state" or lowest-energy solution.

QMC uses two steps

I) Variational Monte Carlo (VMC)

- makes an inspired guess about a parametrized form of the answer
- Can have sub-cluster structure, like α+t+n for ^{8}Li or α+α+α for ^{12}C
- determines best values of parameters
- result is an approximation: Ψ_T

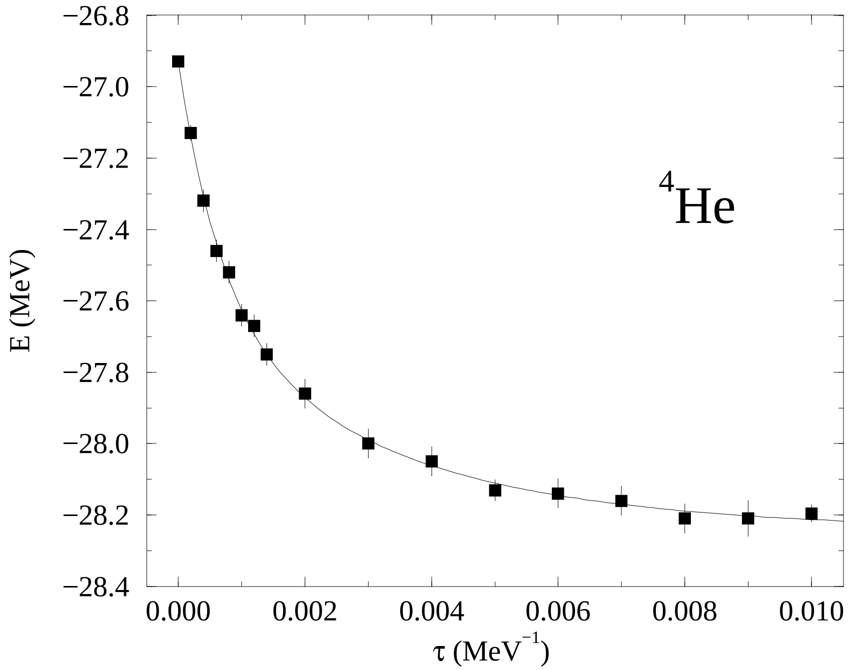
II) Green's Function Monte Carlo (GFMC)

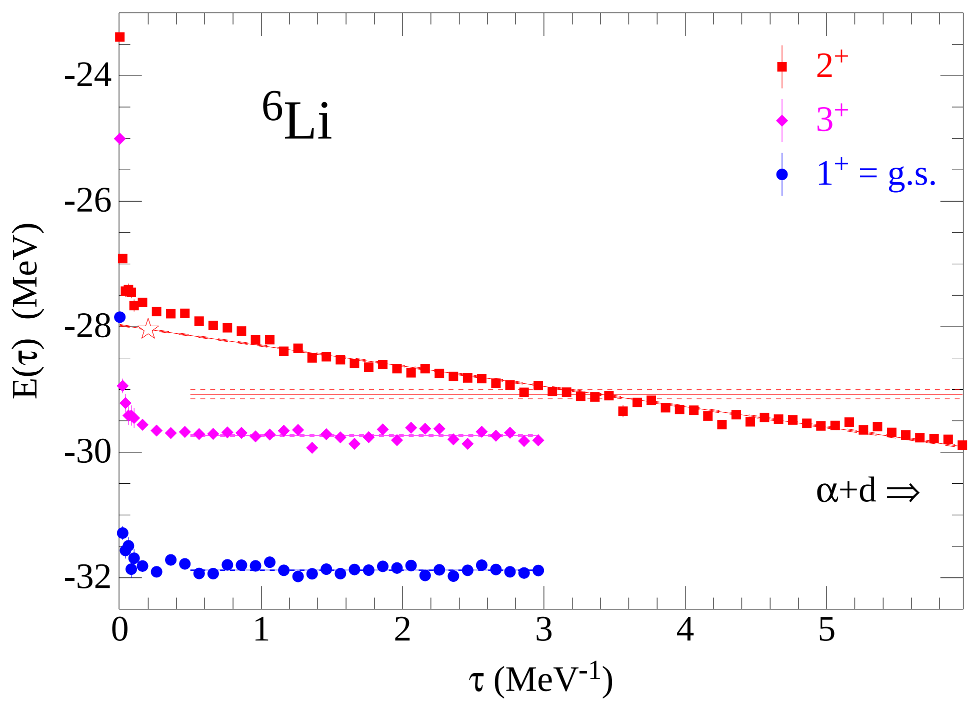
- VMC Ψ_T is propagated (iterated) towards exact solution: $\Psi_n \rightarrow \Psi_0$
- Uses "small-time-step" approximation to single iteration: $G(\mathbf{R}_n, \mathbf{R}_{n-1})$
- Each iteration is another nested $3A$-dimensional integral:

$$\Psi_n(\mathbf{R}_n) = \int G(\mathbf{R}_n, \mathbf{R}_{n-1}) \left[\cdots \left[\int G(\mathbf{R}_2, \mathbf{R}_1) \left[\int G(\mathbf{R}_1, \mathbf{R}_0) \Psi_T(\mathbf{R}_0) d\mathbf{R}_0 \right] d\mathbf{R}_1 \right] \cdots \right] d\mathbf{R}_{n-1}$$

- ^{12}C: typically $1000 \times 3 \times 12$ dimensional integral; done by Monte Carlo
- Monte Carlo samples are killed or replicated in branching random walk – total fluctuates

EXAMPLES OF GFMC PROPAGATION





Curve has $\exp(-E_i\tau)$ with

E_i = 1480, 340 & 20.2 MeV

(20.2 MeV is first ^{4}He 0^+ excitation)

Ψ_T has small amounts of 1.5 GeV contamination

g.s. (1^+) & 3^+ stable after τ = 0.2 MeV^{-1}

2^+ (a broad resonance) never stable – decaying to separated α & d

$E(\tau$=0.2) is best GFMC estimate of resonance energy

Representing Ψ_T in the Computer

$\Psi_T(\vec{r}_1, \vec{r}_2, \cdots, \vec{r}_A)$ is a vector in spin-isospin space
$[2^A$ components for spin $] \times [N_T$ components for isospin $]$

- $N_T = \binom{A}{Z}$ for proton-neutron basis
- $\quad\quad = \frac{2T+1}{A/2+T+1} \binom{A}{A/2+T}$ for good isospin basis

Potentials (v_{ij}, V_{ijk}) and correlations (u_{ij}, U_{ijk}) involve repeated operations on Ψ

$$\sigma_i \cdot \sigma_j = 2(\sigma_i^+ \sigma_j^- + \sigma_i^- \sigma_j^+) + \sigma_i^z \sigma_j^z = 2P_{ij}^\sigma - 1 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 2 & 0 \\ 0 & 2 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \text{ on } \begin{pmatrix} \uparrow\uparrow \\ \uparrow\downarrow \\ \downarrow\uparrow \\ \downarrow\downarrow \end{pmatrix}$$

These result in sparse matrices containing noncontiguous 4×4 and 8×8 blocks

Consider spin part of A=3 w.f.; $\sigma_i \cdot \sigma_j$ will not mix different isospin components:

$$\Psi = \begin{pmatrix} a_{\uparrow\uparrow\uparrow} \\ a_{\uparrow\uparrow\downarrow} \\ a_{\uparrow\downarrow\uparrow} \\ a_{\uparrow\downarrow\downarrow} \\ a_{\downarrow\uparrow\uparrow} \\ a_{\downarrow\uparrow\downarrow} \\ a_{\downarrow\uparrow\downarrow} \\ a_{\downarrow\downarrow\downarrow} \end{pmatrix} \; ; \; \sigma_2 \cdot \sigma_3 \Psi = \begin{pmatrix} a_{\uparrow\uparrow\uparrow} \\ 2a_{\uparrow\downarrow\uparrow} - a_{\uparrow\uparrow\downarrow} \\ 2a_{\uparrow\uparrow\downarrow} - a_{\uparrow\downarrow\uparrow} \\ a_{\uparrow\downarrow\downarrow} \\ a_{\downarrow\uparrow\uparrow} \\ 2a_{\downarrow\downarrow\uparrow} - a_{\downarrow\uparrow\downarrow} \\ 2a_{\downarrow\uparrow\downarrow} - a_{\downarrow\downarrow\uparrow} \\ a_{\downarrow\downarrow\downarrow} \end{pmatrix} \; ; \; \sigma_1 \cdot \sigma_2 \Psi = \begin{pmatrix} a_{\uparrow\uparrow\uparrow} \\ a_{\uparrow\uparrow\downarrow} \\ 2a_{\downarrow\uparrow\uparrow} - a_{\uparrow\downarrow\uparrow} \\ 2a_{\downarrow\uparrow\downarrow} - a_{\uparrow\downarrow\downarrow} \\ 2a_{\uparrow\downarrow\uparrow} - a_{\downarrow\uparrow\uparrow} \\ 2a_{\uparrow\downarrow\downarrow} - a_{\downarrow\uparrow\downarrow} \\ a_{\downarrow\downarrow\uparrow} \\ a_{\downarrow\downarrow\downarrow} \end{pmatrix}$$

Specialized table-driven subroutines carry out these operations

Scaling of Ψ_T Calculation Time With Nucleus

	Pairs	Spin×Isospin	$\prod(/^8\mathrm{Be})$
^{4}He	6	8×2	0.002
^{6}Li	15	32×5	0.048
^{7}Li	21	128×14	0.75
^{8}Be	28	128×14	1.
^{8}Li	28	128×28	2.
^{9}Be	36	512×42	15.
^{10}B	45	512×42	19.
^{10}Be	45	512×90	41.
^{11}Li	55	2048×110	247.
^{12}C	66	2048×132	356. $\longrightarrow$ 500.
^{16}O	120	32768×1430	112,065.
^{40}Ca	780	$3.6\times10^{21}\times6.6\times10^{9}$	5.6×10^{19}

Making It Parallel – Old Method

Master-slave structure

Each slave gets configurations to propagate

Results sent back to master for averaging as generated

During propagation, configurations multiply or are killed

- Work load fluctuates
- Periodically master collects statistics and tells slaves to redistribute
- Slaves have work set aside to do during this synchronization

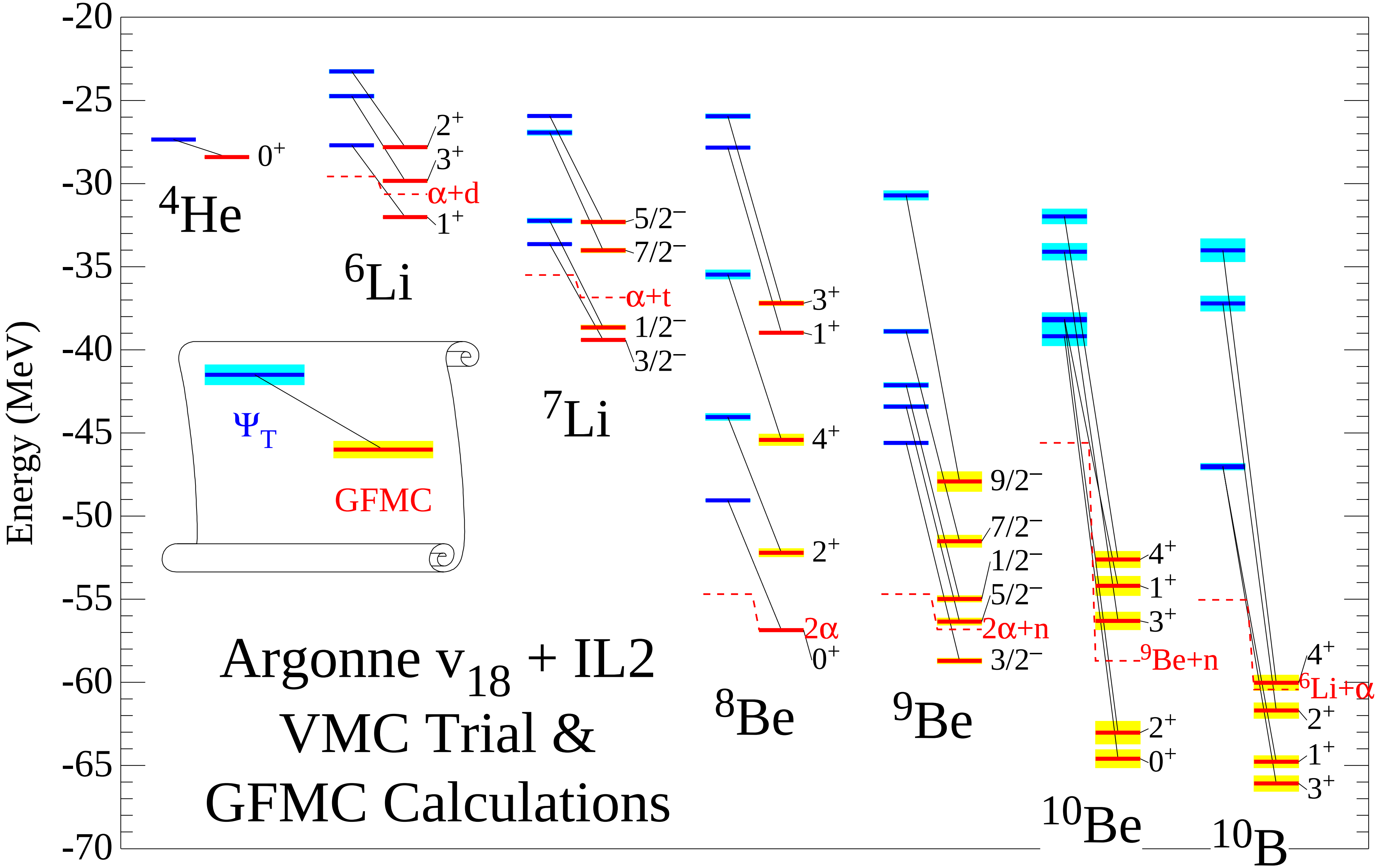
Large calculations have very low (minutes) frequency of communication

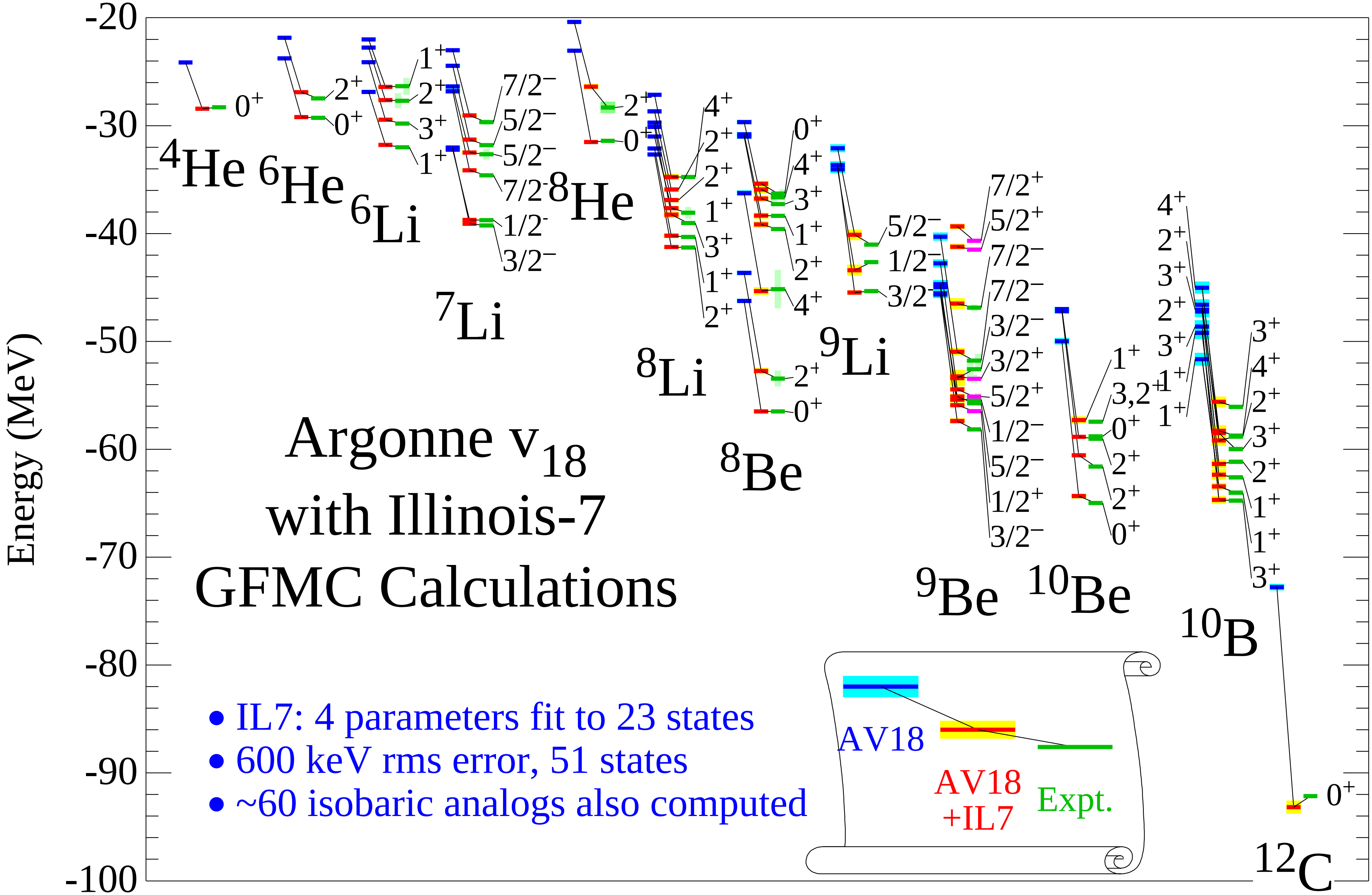
Parallelization efficiencies typically 95%

92% efficency obtained on 2048-processor Seaborg run; 0.55 TFLOPS.

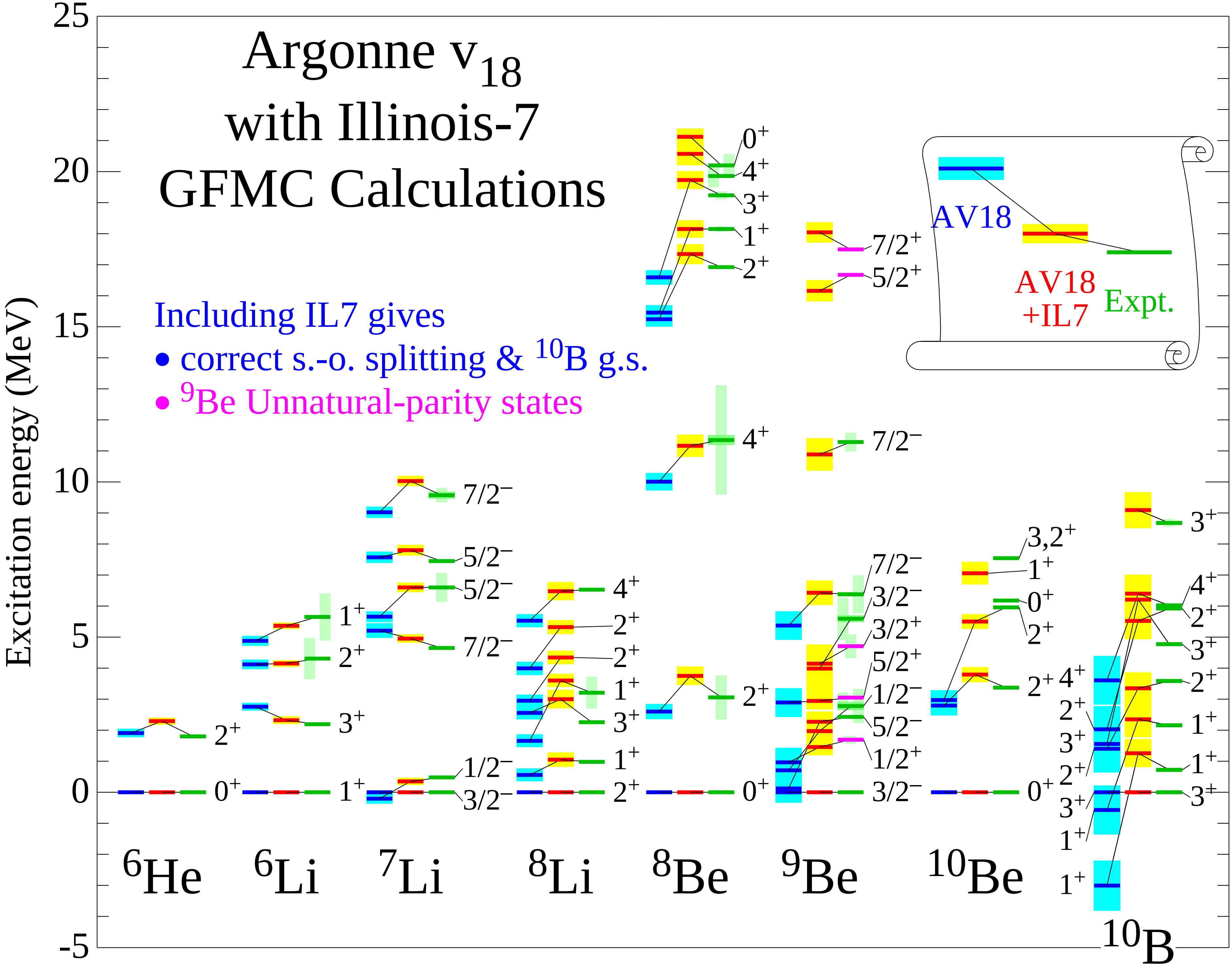
Works well up to 10 nucleons and < 5000 nodes–more Monte Carlo samples than nodes

GFMC makes a big improvement on VMC energies for $A \geq 6$





Energy (MeV)
-20
-30
-40
-50
-60
-70
-80
-90
-100
^{4}He
^{6}He
^{6}Li
^{7}Li
^{8}He
^{8}Li
^{8}Be
^{9}Li
^{9}Be
^{10}Be
^{10}B
^{12}C
Argonne v_{18}
with Illinois-7
GFMC Calculations
• IL7: 4 parameters fit to 23 states
• 600 keV rms error, 51 states
• ~60 isobaric analogs also computed
AV18
AV18
+IL7
Expt.



Argonne v_{18}
with Illinois-7
GFMC Calculations
Including IL7 gives
• correct s.-o. splitting & ^{10}B g.s.
• ^{9}Be Unnatural-parity states
AV18
AV18 +IL7
Expt.
Excitation energy (MeV)
^{6}He
^{6}Li
^{7}Li
^{8}Li
^{8}Be
^{9}Be
^{10}Be
^{10}B

SECOND 0^+ (HOYLE) STATE OF ^{12}C

The Second 0^+ state of ^{12}C is the famous triple-alpha burning or Hoyle state

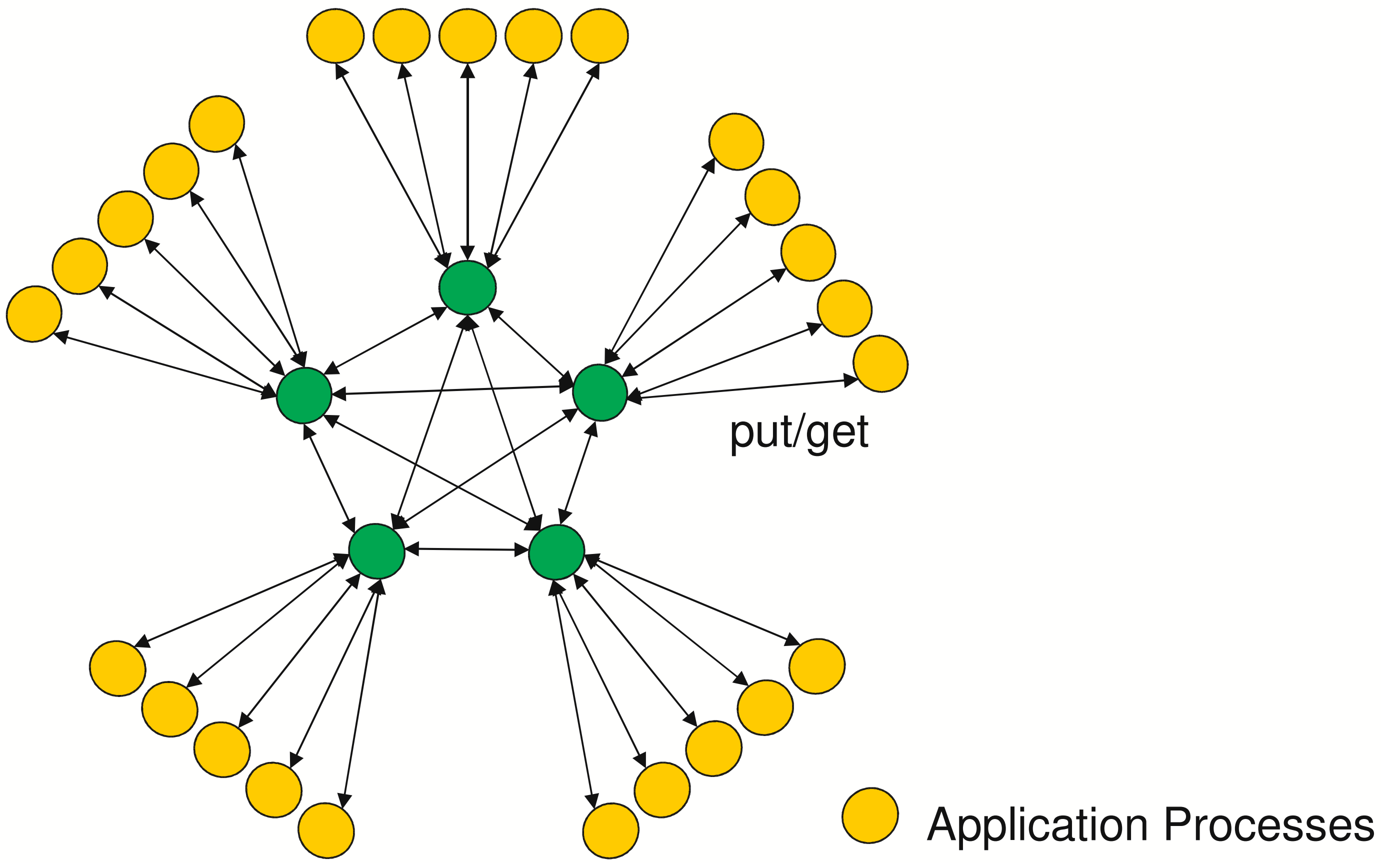
- Resonance only 0.38 MeV above 3α breakup threshold
- Doorway state postulated by Fred Hoyle for $3\alpha \rightarrow {}^{12}$C in stars. (Without the Hoyle state, there would be almost no carbon, and hence I would not be giving this lecture.)
- Shell model calculations show it to be 4-particle 4-hole excitation
- It is one of the goals of the UNEDF SciDAC
- Not yet converged in *ab initio* no-core shell model
- Our trial wave functions should have the necessary flexibility (triple-α structures)
- Need to make many calculations to explore this
- These are BG/P class calculations
- Want fewer Monte Carlo samples than nodes
- Need finer-grain parallelization than previously
- Automatic Dynamic Load Balancing (ADLB) library developed as the answer

AUTOMATIC DYNAMIC LOAD BALANCING – THE VISION

Being developed by Rusty Lusk and Ralph Butler

- Explicit master not needed:
  - Slaves make calls to ADLB library to off-load or get work
  - ADLB accesses local and remote data structures (remote ones via MPI)
- Simple Put/Get interface for application code hides most MPI calls
  - Advantage: multiple applications may benefit
  - Wrinkle: variable-size work units introduce some complexity in memory management
- Proactive load balancing in background
  - Advantage: application never delayed by search for work from other slaves
  - Wrinkle: scalable work-stealing algorithms not obvious

AUTOMATIC DYNAMIC LOAD BALANCING – WORK FLOW



Automatic Dynamic Load Balancing – The API

- Startup and termination
  - ADLB_Init(num_servers, am_server, app_communicator)
  - ADLB_Server()
  - ADLB_Set_No_More_Work()
  - ADLB_Finalize()
- Putting work or answers
  - ADLB_Begin_Batch_Put(common_buffer, length) – optional
  - ADLB_Put(type, priority, length, buffer, answer_destination)
  - ADLB_End_Batch_Put() – optional
- Getting work or answers
  - ADLB_Reserve(req_types, work_handle, length, type, priority, answer_destination)
  - or ADLB_Ireserve($\cdots$)
  - ADLB_Get_Reserved(work_handle, buffer)

ADLB – Current GFMC Implementation

Old GFMC

Each slave gets several configurations

Slave

propagates configurations

(few w.f. evaluations)

replicates or kills configs (branching)

$\longrightarrow$ periodic global redistribution

computes energies

(many w.f. evaluations)

Need $\sim$10 configs per slave

^{12}C will have only $\sim$10,000 configs.

Can't do on more than 2000 processors

Configurations cannot be unit of parallelization

With ADLB

A few "boss" slaves manage the propagation:

- Generate propagation work packages
  - – Answers used to make 0,1,2,$\cdots$ new propagation packages (branching)
  - – Number of prop. packages fluctuates
  - – Global redistribution may be avoided
- Generate energy packages – No answers

When propagation done, become worker slaves

Most slaves ask ADLB for work packages:

- Propagation package
  - – Makes w.f. and $3N$ potential packages
- Energy package
  - – Makes many w.f. packages
  - – Makes $3N$ potential packages
  - – Result sent to Master for averaging
- Wave Function or $3N$ potential package
  - – Result sent to requester

Wave function is parallelization unit

Can have many more nodes than configurations

Example of ADLB Coding – The Kinetic Energy Calculation

Kinetic energy requires $6A$ wave functions at small steps from central location

These are farmed out as work packages (WP) via ADLB

(ADLB argument lists are schematic)

1) Make all the WP
Each has a unique key
and position
Put the WP to ADLB

```
do i = 1, A ; do ixyz = 1, 3 ; do is = 1, 2
 key = 100*i + 10*ixyz + is
 xyz = ...
 call ADLB_PUT( [key, xyz], len, &
 & ANY_DEST, my_rank, wf_type, ... )
enddo ; enddo ; enddo
```

2) Get all w.f.

Get answer or work to avoid deadlocks

Process computed w.f.

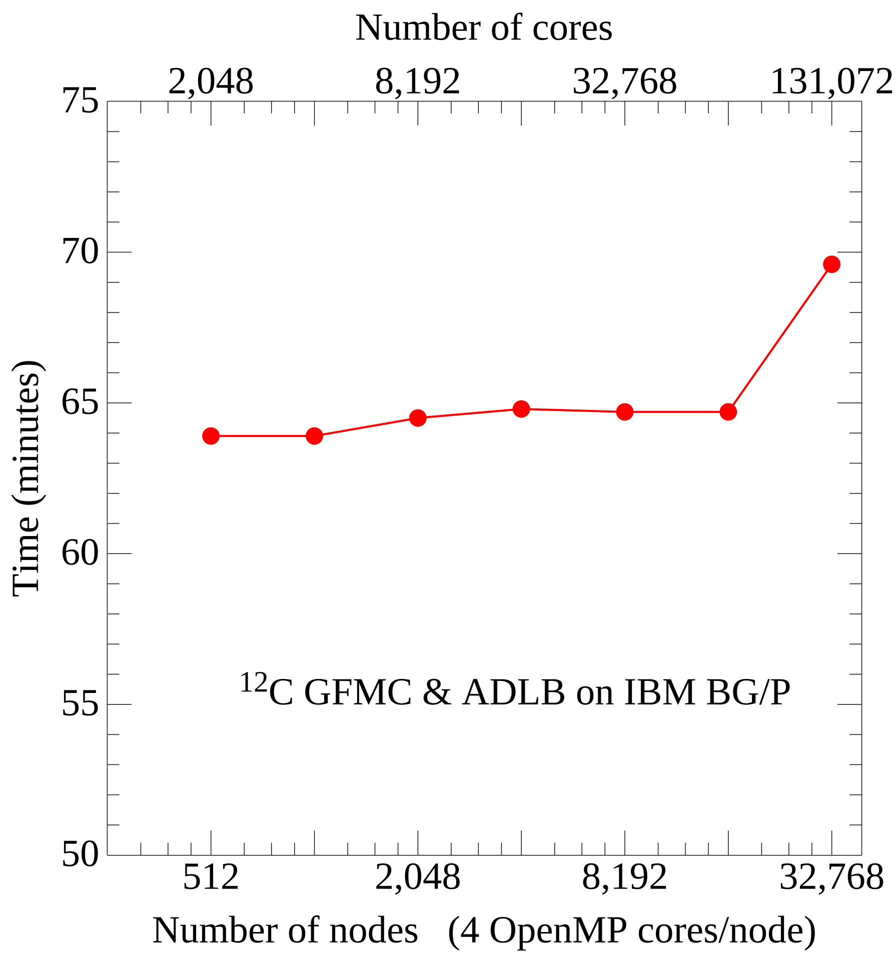
Send answer to the originating rank

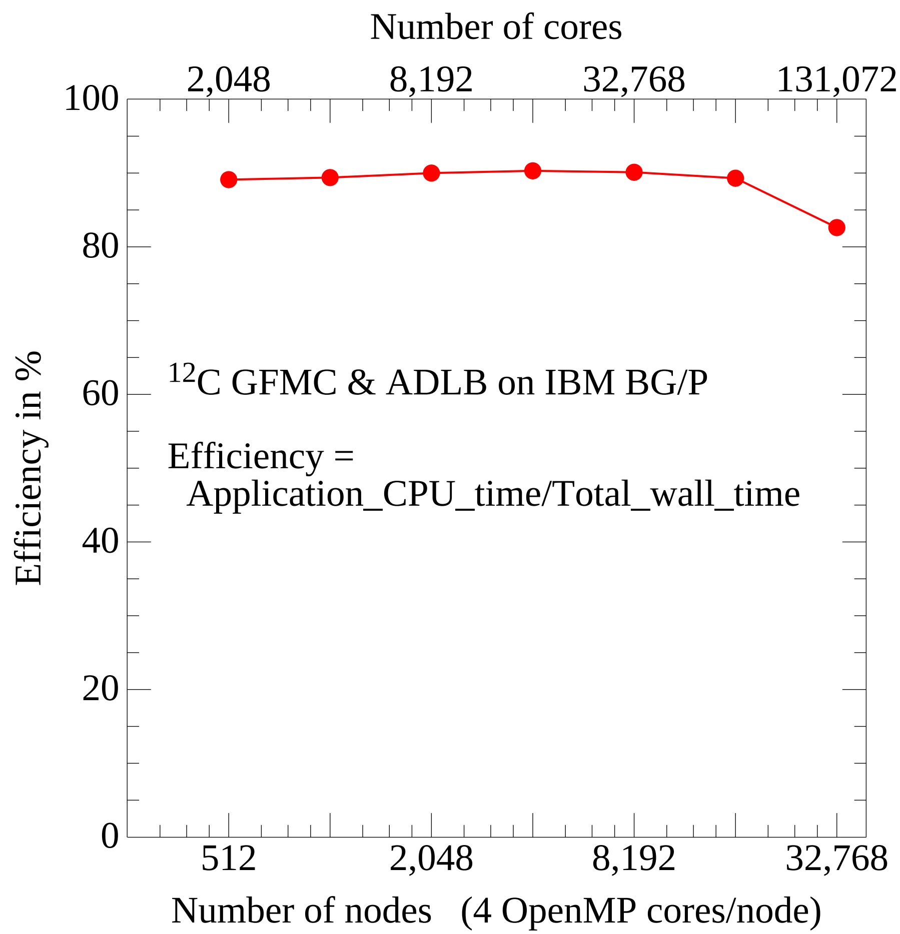
```
num_got = 0
do while ( num_got < 6*A )
 call ADLB_RESERVE( (/wf_ans, wf_type, -1/),&
& type, handle, ans_rank ... )
 if ( type == wf_ans ) then
  call ADLB_GET_RESERVED( ans, handle, ... )
  ...  use key to process
  num_got = num_got + 1
 else
  call ADLB_GET_RESERVED( work, handle, ... )
  compute wave function ...
  call ADLB_PUT( [key, w.f.], &
  & len, ans_rank, ...)
 endif

enddo
```

GFMC Performance Using ADLB on Argonne's IBM BG/P

Weak scaling study – 2 Monte Carlo samples per node

ADLB performance is very good up to 32,768 nodes (131,072 cores)





ADLB is a general purpose library; give it a try! – http://www.cs.mtsu.edu/∼rbutler/adlb

Let us know your experiences with it – spieper@anl.gov

One More Problem – Memory

- Each node has 2 gigabytes of RAM and 4 processors (cores)
  - Only 500 megabytes per core
  - Not enough for ^{12}C if each core is a separate MPI/ADLB node
- Use Open MP to let the 4 cores work as one 2-gigabyte node
  - Directives added to source state which loops can be done in parallel
  - Iterations of the loops must be independent
  - Variables and arrays must be designated as "private" or "shared"
    * Open MP makes multiple copies of private variables
    * Programmer must guarantee non-overlapping stores into shared arrays
- IBM BGP OpenMP is quite successful – speedups of 2.6–3.9 from 4 cores
- MPI/ADLB between nodes with OpenMP on nodes is an example hybrid parallelization

OpenMP changes

Generally only OpenMP directives needed to be added.

One case was more complicated

First Attempt

```
!$OMP PARALLEL DO ...
do n = 1, ...

  do m = ...
   do k = ...
    do ...  many!
     stuff ...
     i = func_1(n,m,k,..)
     j = func_2(n,m,k,..)
!$OMP CRITICAL
     z(i,j) = z(i,j)+..
!$OMP END CRITICAL
    enddo
   enddo
  enddo
enddo
!$OMP END PARALLEL DO
```

This did not speedup well

Final Version

```
!$OMP PARALLEL DO ...
do n = 1, ...
!$  ith = omp_get_thread_num()
  do m = ...
   do k = ...
    do ...  many!
     stuff ...
     i = func_1(n,m,k,..)
     j = func_2(n,m,k,..)

     y(i,j,ith) = y(i,j,ith)+..

    enddo
   enddo
  enddo
enddo
!$OMP END PARALLEL DO
!$OMP WORKSHARE
z(:,:,1) = y(:,:,1)+y(:,:,2) &
&         +y(:,:,3)+y(:,:,4)
!$OMP END WORKSHARE
```

This worked well

OpenMP for work on one node

^{12}C times (seconds) for key subroutines

Subroutine	Seconds		speed up
	No OMP	4 threads	
Wave function	15.7	5.4	2.9
Prop. update	5.5	2.1	2.6
V_{ijk}	29.3	7.5	3.9
Weighted average	18.2	5.5	3.3

Full iteration times (minutes on 512 nodes for 1000 samples)

	Minutes		speed up
	No OMP	4 threads	
Wall time	192.	61.	3.2
CPU time	85,700.	26,783.	3.2

^{12}C RESULTS

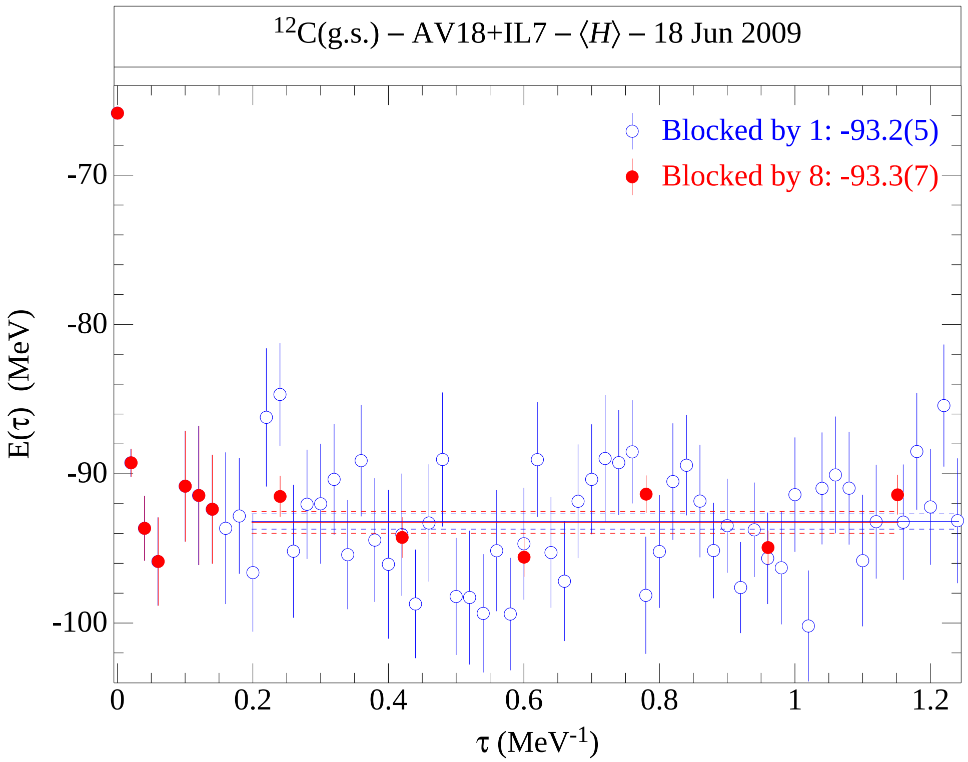
In Dec. 2008 & Jan. 2009, the first ADLB+GFMC calculation of the ^{12}C(gs) was made.

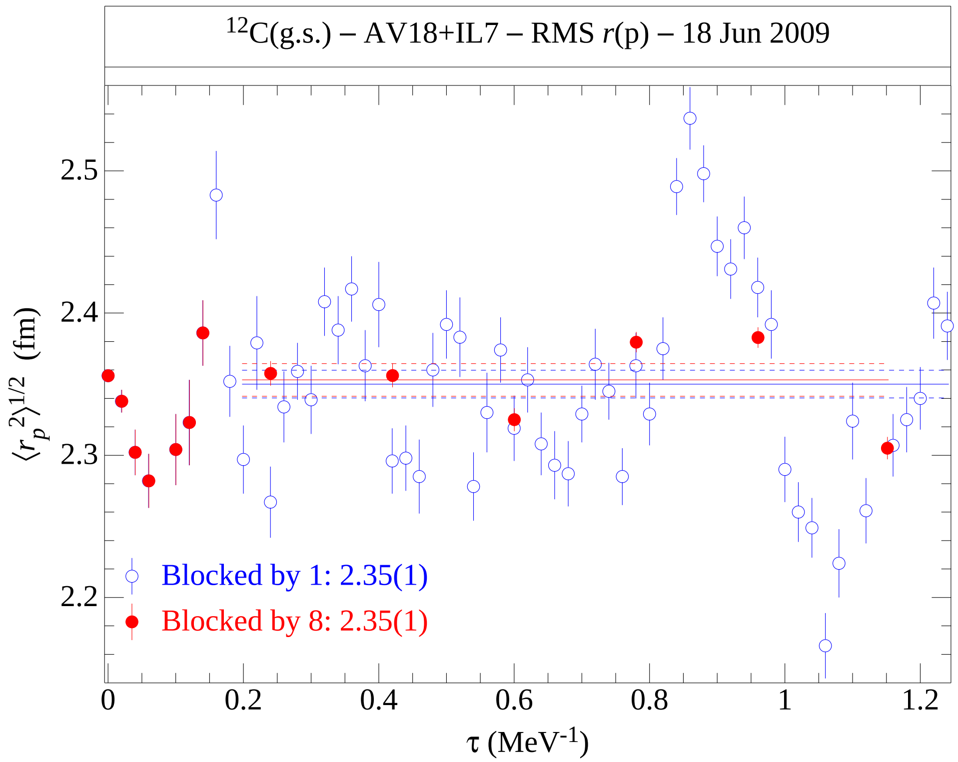
- AV18+IL7 Hamiltonian
- Improved (and slower) Ψ_T than in our previous [approximate] ^{12}C(gs) calculations
- GFMC energy changed only a little from our previous results
- 16,000 configurations propagated to $\tau = 1.24$ MeV^{-1} (2480 steps)
- 40 unconstrained time steps used before energy evaluations
- Used 8,192 nodes (32,768 cores) of BG/P (300,000 processor hours)
- 14 runs for total of 93 hours (first few very short)
- Speed of Ψ_T calculation has been significantly improved since then
- Convergence is very good and shows that
  - smaller maximum τ can be used
  - fewer unconstrained time steps, and hence fewer configurations, can be used

Calculations using Argonne v_{18} & the benchmark modified SSCC v_8' NN potentials without V_{ijk} have also been made

^{12}C Results – Energies & Radii

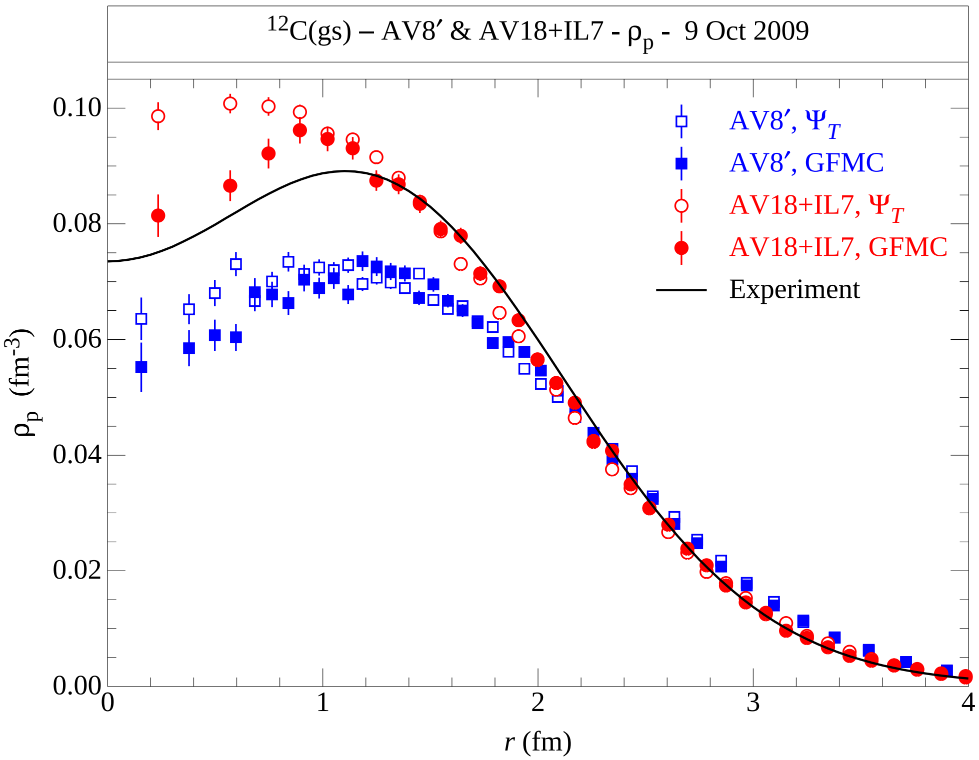
Convergence as a function of GFMC propagation steps or imaginary time (τ)

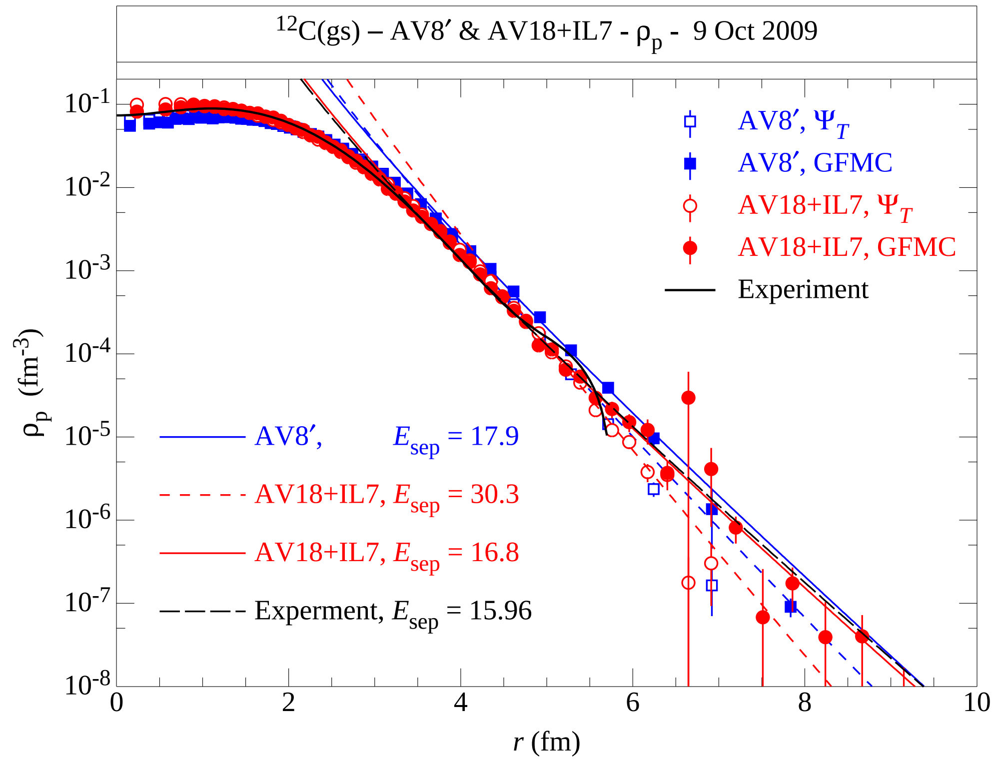




	Energy			RMS radius		
	VMC	GFMC	Expt.	VMC	GFMC	Expt.
AV18	–51.5(2)	–72.8(3)		2.48	2.51	
AV18+IL7	–65.8(2)	–93.2(6)	–92.16	2.36	2.35	2.33

^{12}C RESULTS – ONE-BODY DENSITY





The Ψ_T density is significantly improved by GFMC

- Central dip is generated
- AV18+IL7 tail falls at rate dictated by $E_{\rm sep} = E(^{11}\mathrm{B}) - E(^{12}\mathrm{C})$ instead of twice as fast

Conclusions & Future

We have made much progress in quantum Monte Carlo calculations of light nuclei

- $1-2\%$ calculations of A = 6 – 12 nuclear energies are possible
- Illinois V_{ijk} give average binding-energy errors ≈ 0.6 MeV for $A = 3-12$
- ADLB library with OpenMP allows efficient use of 100,000 processors for GFMC
- Ground state of ^{12}C is well reproduced
- Scattering calculations work well

and there is still much to do

- More ^{12}C including 2^{nd} 0^+ (Hoyle) state
- Lots of scattering states and reactions to be done
  – All big-bang, solar pp chain, & some r-process seeding reactions are accessible.
- GFMC calculations of other properties of nuclei
- Further development of ADLB
  - Current version saves work packages on ADLB servers
    - for ^{12}C, 6% of the nodes are used for this and are unavailable for computing
  - Now working on a version that stores work on all the clients
    - One-sided puts and gets used to move work packages
    - Only one ADLB server to control things
  - Will be working towards the next generation Blue Gene

To Learn More

Pointers to the following are at http://www.phy.anl.gov/theory/staff/SCP.html & RBW.html

- *Nucleon-nucleon interactions*, R. B. Wiringa, in Contemporary Nuclear Shell Models, ed. X.-W. Pan, D. H. Feng, and M. Vallières (Springer-Verlag, Berlin, 1997)
- *Monte Carlo calculations of nuclei*, S. C. Pieper, in Microscopic Quantum Many-Body Theories and Their Applications, ed. J. Navarro and A. Polls, Lecture Notes in Physics **510** (Springer-Verlag, Berlin, 1998)
- *Quantum Monte Carlo Calculations of Light Nuclei*, S. C. Pieper and R. B. Wiringa, Annu. Rev. Nucl. Part. Sci. **51**, 53-90 (2001)
- *Quantum Monte Carlo Calculations of Light Nuclei*, S. C. Pieper, in Proceedings of the ”Enrico Fermi” Summer School, Course CLXIX, ed. A. Covello, F. Iachello, and R. A. Ricci (Società Italiana di Fisica, Bologna, 2008); arXiv:0711.1500 [nucl-th]
- A simplified VMC program and description: *Variational Monte-Carlo Techniques in Nuclear Physics*, J. A. Carlson and R. B. Wiringa, Computational Nuclear Physics 1, ed. K. Langanke, J. A. Maruhn, and S. E. Kain (Springer-Verlag, Berlin, 1990), Ch. 9
  source & input files available at http://www.phy.anl.gov/theory/research/vmc-demo
- ADLB load-balancing library is at http://www.cs.mtsu.edu/∼rbutler/adlb